Historic, archived document

Do not assume content reflects current scientific knowledge, policies, or practices.

ERS 122
A281.9
AG83E Reserve

USING DISCRETE PROGRAMMING

By Clark Edwards

Reprinted from Agricultural Economic Research, April 1963, Vol. XV, No. 2 by the Economic and Statistical Analysis Division, ERS, May 1963

ERS-122

MAY 1963

Tri-Agency Reading Room
Date_____
Room 505
500 12th St., SW
Washington, D. C. 20250

Using Discrete Programming

By Clark Edwards

In many important economic problems, a variable is maximized subject to constraints. In a subset of such problems, a linear combination of decision variables is maximized subject to linear constraints. The latter subset is amenable to linear programming analysis. Efforts to expand the usefulness of linear programming methods usually involve incorporating nonlinear elements either in the criterion function or in the constraints. Such efforts frequently result in discovering ways to incorporate the nonlinear element in some acceptable linear form, thus retaining the usual linear programming procedure but broadening the researcher's capacity to apply the method to important economic problems (*9*).[1] Discrete programming is a case in point. Discrete programming problems and ordinary linear programming problems are about the same, except that a side condition is imposed that some of the decision variables must take on discrete values, usually nonnegative integers. The resultant, noncontinuous nature of the criterion function or of the constraints places discrete programming in the class of nonlinear programming (*10*). Sufficient conditions for a solution to discrete programming problems have been known for several years (*15*). Recently, systematic procedures for solving discrete programming problems have been put forward (*14, 16*). This paper discusses one of them. Decks and tapes for solving such problems on high-speed computers are not yet abundant, but it would be easy to supply them should the demand arise.

A LINEAR PROGRAMMING problem is one in which elements of a decision vector x are to be chosen in such a way as to maximize Q

(1) $Q=c'x$

subject to

(2) $Ax \leqq b$

and to

(3) $x \geqq 0$

[1] Italic numbers in parentheses refer to Literature Cited and Selected References, p. 59.

Integer programming is done under the additional constraint that

(4) some elements of x are integers.

If Q is net revenue in a farm management problem and x_j is the swine activity, to insist by (4) that x_j be an integer is simply to require that if some sows are to be farrowed in the optimal farm plan, an integral number must be farrowed. If, without the discrete restriction, the optimal number of sows were estimated at 17.682, the question arises (in this case, perhaps, a somewhat trivial question) as to whether some limited resources should be withdrawn from other activities to increase sow numbers to 18 (or 19) or whether it would be more profitable to farrow 17 (or 16) sows and release some limited resources for other uses.

Some integer programming restrictions are shown in figure 1. Line ABCD represents the feasible, noninteger boundary for pairs of x_1 and x_2 given levels for activities x_3 to x_n. Inside ABCD, the boundary for integer pairs of values is marked with a dotted line. With RR depicting the price ratio, or choice indicator, B marks the high profit point of the noninteger solution. E marks the high profit point for x_1 and x_2 integers but x_3 to x_n held constant. In a simultaneous integer solution of all n activities, the optimal level of, say, x_7 might change sufficiently that the best combination of x_1 and x_2 would shift away from E, say, to F or G.

The marginal utility of adding a discrete restriction to a programming problem may be negligible when the magnitude of the decision variable is large relative to the size of the discrete jump. For example, it may be less interesting to know if a number like 103.467 should round up to 104 or down to 103 than to know if a number like 0.674 should round up to 1.0 or down to zero. It is the systematic way in which discrete programming handles relatively large jumps that makes it useful and interesting.

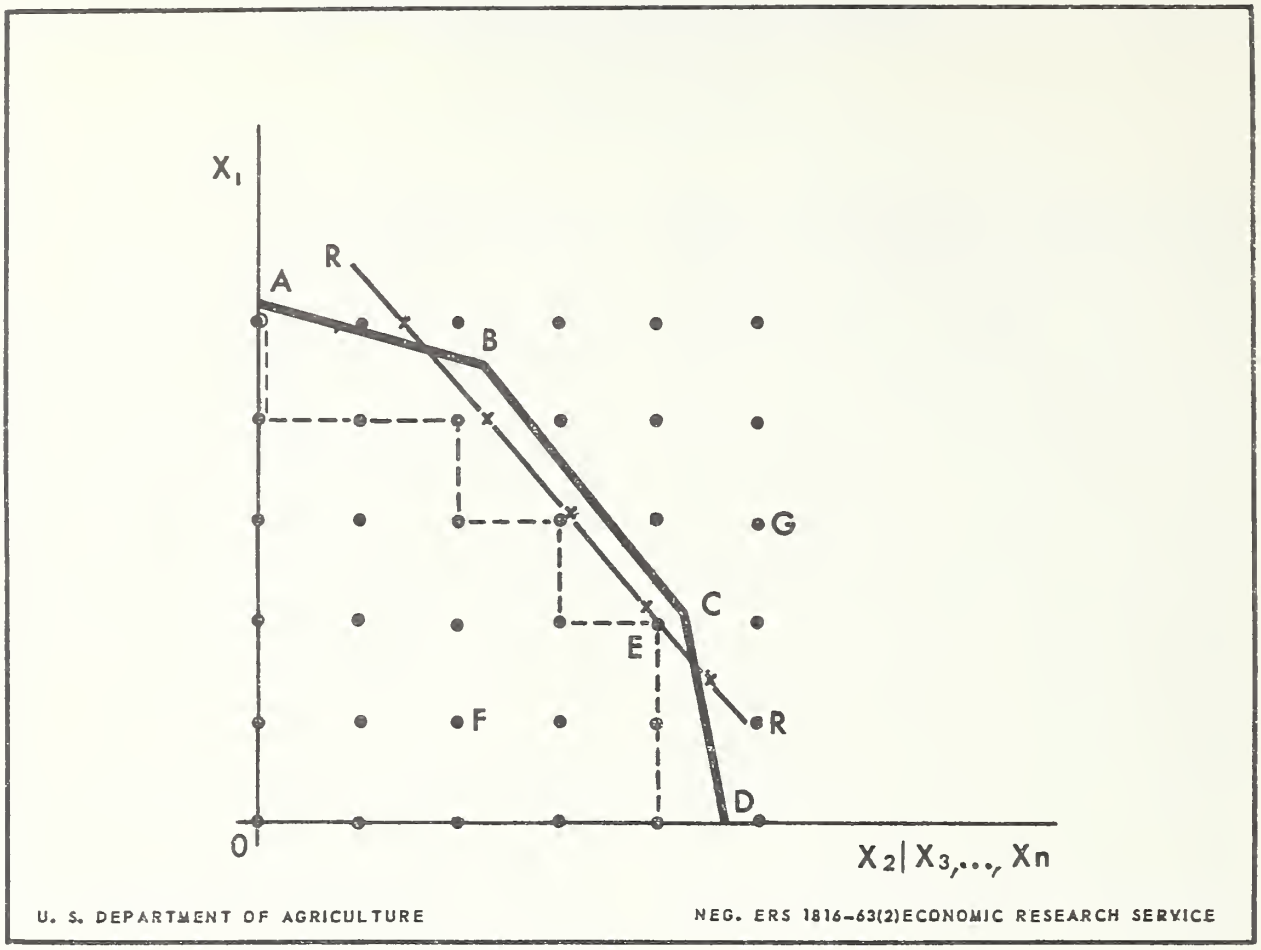

FIGURE 1.—Some integer programming restrictions.

The integer programming method can handle several kinds of problems (*8*):

1. It may ensure that variables in the solution are positive integers.

2. It may incorporate all-or-nothing-at-all restrictions such as owning a combine or not; and as using the full wheat acreage allotment or producing no wheat. Lump-sum costs or resource requirements associated with the all-or-nothing decision may be allowed for. This sort of restriction usually employs a variable confined to two values, 0 and 1.

3. Admissible ranges for levels of inputs or outputs may be established by integer programming, as when a poultry enterprise must have at least 1,000 layers if poultry is included in the optimal farm plan. Variations in net revenues or resource requirements with respect to ranges of values for specific variables may be incorporated.

4. Either-or choices may be analyzed in the integer programming framework as when one would either milk dairy cows or feed beef cows, but not both.

5. Multiple choice problems in which one element is chosen from a set of elements are amenable to integer programming. Choosing among equidistant numbers on the real line is a trivial example and includes choosing among real integers, even numbers, numbers which are multiples of 10, and so on. Choosing among a finite set of numbers not equidistant from one another is associated with some decisions concerning size of enterprise or level of resource use. In an economies-of-scale study, one could analyze a feeder cattle enterprise with the following alternative sizes: 0, 25, 100, 500, and 1,000. Choices among finite sets not usually measured numerically may also be incorporated in integer programs such as choosing

among five crop rotations considered for a farm; or, choosing among three feeding methods considered for a hog enterprise.

Writing the Discrete Restrictions

Writing a discrete restriction in a programming problem frequently involves devising a scheme for restating a nonlinear, discontinuous problem in a linear, continuous form, equations (1), (2), and (3), and then imposing the added restriction that one or more variables are integers, equation (4). Some examples follow:

Example 1.—Let x_j be acres of corn and x_k a (0,1) variable representing a fixed complement of machinery and equipment necessary for handling a corn crop. Net revenue, equation (1), includes the terms

$$c_j x_j + c_k x_k \tag{5}$$

where c_k is the annual cost of making the machinery and equipment available and c_j is the change in net revenue associated with growing an acre of corn given that the machinery has been made available.

A complete statement of the corn-machinery problem could include the statement that

$$\begin{aligned} x_j > 0 &\rightarrow x_k = 1 \\ x_k = 0 &\rightarrow x_j = 0 \end{aligned} \tag{6}$$

which says that if corn is grown, machinery must be made available, and if machinery is not made available, corn can not be grown. That machinery might be made available without growing corn is feasible according to (6) but not economic according to (5). Growing corn without equipment is not feasible according to (6).

While (6) states the discrete restriction, it is not in the form of equation (2) and is not amenable to ordinary programming procedures. The restriction may be translated to a usable form by introducing a dummy parameter based on the knowledge that the variable x_j has an upper bound imposed by other constraints to the problem. For instance, one of the restrictions in (2) may state that only 100 acres of corn land are available in which case any real number not less than 100 may be used as an upper bound for x_j. Let us call the dummy parameter α_j. Now we write the restriction

$$\begin{aligned} x_j &\leqq \alpha_j x_k \\ x_k &\text{ an integer} \end{aligned} \tag{7}$$

which conforms to (2) and (4) and can be shown to satisfy (6). The terms (5) would appear in equation (1) and the inequality in (7) would appear as one of the equations in (2). According to (7), it is not feasible to grow corn unless machinery is made available. If corn is not grown on the farm, the profit maximizing criterion requires that no machinery is made available. If corn is grown, then (7) requires that at least one set be made available and the profit maximizing criterion requires that not more than one set of machinery is acquired. Thus x_k will be either 0 or 1 in the optimal solution. Many enterprises require an initial, overhead cost in addition to a variable, unit cost. Discrete programming is an efficient way to distribute fixed charges. The term $c_k x_k$ in (5) represents a fixed charge in terms of a cash outlay, or a lump sum reduction in net revenue.

If the fixed cost were measured in terms of using up a lump sum of limited resources such as building space or labor, rather than in terms of directly reducing net revenue, a similar restriction results. Interpreting (7) as above, set $c_k = 0$ to show that net revenue is not directly affected by a change in the (0,1) variable, x_k. Then, letting the ith equation in (2) represent the labor restriction, let a_{ik} reflect the fixed labor requirement. See appendix table 1, line 5, for an illustration. Now if $x_k = 1$ the fixed labor supply is reduced by a_{ik}, but if $x_k = 0$ the labor supply is not affected.

Example 2.—In example 1, the discrete variable takes one of two values: 0 and 1. In other problems, the discrete variable may assume any nonnegative integer. For instance, perhaps the number of storage bins is to be an integer and there must be at least one bin for each 10,000 bushels of grain; or the number of machines employed in an activity must be an integer and there must be at least 1 machine for each 10 units of labor.

Let x_j be an activity providing labor and x_k one providing machines and suppose the number of machines supplied must be an integer. The supplies of x_j and x_k may be perfectly elastic with unit costs represented by c_j and c_k. In this event, neither variable would have an upper bound as did the variable in the previous example. Appropriate coefficients in the labor restriction equation would ensure that an adequate quantity of labor,

x_j, is made available. In addition, an equation saying that $x_j \leqq \alpha x_k$ as in equation (7), will ensure that there will be at least one machine for each α units of labor. Hence, while other restrictions in (2) would lead to the optimal labor utilization, (7) would ensure an adequate supply of machinery.

Example 3.—Sometimes it is desirable to impose the restriction that if an activity is used, it is used at least at a minimal level. For instance, if wheat is to be grown then at least 15 acres must be grown; and, if cows are to be milked then at least 25 head must be in the herd. Let us say that for a dairy activity, x_j, it is known from other constraints in the problem that 100 is an upper bound for x_j. Furthermore, we know from prior economic analysis that if there is to be a dairy activity in the final solution, the herd must have at least 25 cows. That is, we wish to impose

$$\begin{array}{ll} & \text{either } 25 \leqq x_j \leqq 100 \\ (8) & \\ & \text{or} \qquad x_j = 0 \end{array}$$

This nonlinear, discontinuous condition can be written in a form suitable for ordinary programming methods by writing

$$\begin{array}{ll} & x_j \leqq \alpha_j x_k \\ (9) & x_j \geqq \beta_j x_k \\ & x_k \text{ an integer} \end{array}$$

where α_j is the upper bound, 100, and β_j is the lower bound, 25. If introducing the first 25 cows involved an overhead cost, this could be reflected by a value for c_k in the profit equation. If the first 25 cows affected the ith restriction differently from the additional cows, as by requiring a lump sum labor requirement in addition to the variable labor requirement per cow, this could be reflected by a value for a_{ik} in the ith equation of (2). In (9), it follows that if $x_k = 0$, $x_j = 0$. If $x_k = 1$, $25 \leqq x_j \leqq 100$ as required in (8). An uneconomic-size herd of, say, 10 head is not feasible because $x_j = 10$ violates the first inequality of (9) if $x_k = 0$ and violates the second inequality if $x_k = 1$.

Example 4.—Discrete variables may assume one of several values which are not equidistant from one another. One might want to consider wheat acreages of 0, 15, and 40 acres or consider dairy enterprises of 0-, 25-, 60-, and 100-cow herds without examining other activity levels. In the dairy example, each herd size may be considered efficient for alternative sets of buildings and equipment, qualities of cows, rates of feeding, and requirements of labor.

A restriction of the form

$$\begin{array}{ll} & x_j = 25x_k + 60x_m + 100x_n \\ (10) & \\ & x_k,\ x_m,\ x_n \text{ are integers} \end{array}$$

would permit examination of the four dairy alternatives. If each of the integer variables were zero, x_j would also be zero.

If, in the profit maximizing process, more than one herd appeared in the final solution and at most one herd was considered admissible, the further restriction would be imposed that

$$1 \geqq x_k + x_m + x_n \qquad (11)$$

Forcing the equality in (11) would result in exactly one herd and allowing the inequality would allow none. Frequently, an equation of type (11) will not be violated even when not explicitly imposed due to the nature of the criterion function (1) and to other restrictions in (2).

As in prior examples, values for c_k, c_m, c_n, a_{ik}, a_{im}, and a_{in} may be incorporated to account for lump sum increments in costs, returns, or resource use associated with alternative levels of x_j.

Example 5.—If a farm organization could include either Herefords or Holsteins but not both, farrow either one litter of pigs per year or two litters but not both, or plant either corn or soybeans in a field but not both, then a restriction of the form

$$x_i \cdot x_j = 0 \qquad (12)$$

would require that at least one of the activities not enter into the solution. This nonlinear condition may be imposed by two linear inequalities in an integer program. Let α_i and α_j be upper bounds for x_i and x_j, respectively. Such bounds are always implied by other constraints in (2). Then incorporate the restriction that

$$\begin{array}{ll} & x_i \leqq \alpha_i x_k \\ (13) & x_j \leqq \alpha_j (1 - x_k) \\ & x_k \text{ an integer} \end{array}$$

with the understanding that x_k is confined to the integers 0 and 1. Should difficulties in computation arise from $x_k \geqq 2$, simply impose the additional restriction that $x_k \leqq 1$. See appendix table 1, lines 6 and 7, for an illustration.

Displaying the Discrete Solution

Let the integer programming problem be stated in the form of equations (1) to (4) with (2)

including equations such as (7), (9), (10) and (13). The first step in displaying the integer solution is to solve equations (1), (2), and (3), ignoring (4) to get a first approximation (*14*).

The first approximation allows variables which are supposed to be integers to take on noninteger values and therefore need not be a feasible solution. For example, one of the (0,1) variables might have the value of .90 in the first approximation. This might mean that only 90 percent of a combine or 90 percent of a dairy barn is allowed for in the solution. The next step is to add an additional equation to the noninteger solution matrix, incorporating information as to which of the variables are to be integers, and to proceed with the computations.

The rules for discrete programming are simpler if equation (4) is changed to read "all x_j are integers." Let us examine first the complete integer solution and then take up the partial integer solution.

The Complete Integer Solution

Let equation (14) represent the ith of m restrictions in the first approximation, or the noninteger solution of a linear program for which an integer solution is desired.

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \qquad (1 \leq i \leq m) \tag{14}$$

where it is understood that

$$\begin{aligned} a_{ij} &= 1 && \text{for the coefficient of the } x_j \text{ having the value } b_i \text{ in the current basis, say } x_k \\ a_{ij} &= 0 && \text{for coefficients of all other } x_j \text{ included in the current basis} \\ -\infty < a_{ij} &< \infty && \text{for coefficients of all } x_j \text{ not included in the current basis} \end{aligned} \tag{15}$$

Some of the x_j in (14) are real activities, some are slack variables and others were introduced through equations such as (7). Any of the m equations may be chosen for (14) provided b_i is not, in fact, an integer in the first approximation. See appendix table 1 line 12 for an example.

The assertion that all x_j are integers requires that all b_i in (14) are integers. But (14) was derived without imposing the integer restriction. If, on inspection, the b_i are integers, there is no further integer problem. If some b_i are not integers, an additional constraint is required. The additional constraint will expand the solution matrix from an m by n to an m+1 by n+1 matrix. The additional constraint is devised by operating on equation (14).

Let $[b_i]$ equal the largest integer less than or equal to b_i and define δ_i such that

$$\delta_i = b_i - [b_i] \tag{16}$$

Substituting (14) into (16),

$$-\sum_{j \neq k}^{n} a_{ij}x_j + [b_i] - x_k = -\delta_i \tag{17}$$

where x_k is the activity in the current basis assigned the value b_i.

Imposing the condition that all x_i are integers on (17) implies that a change in any a_{ij} by an integer amount changes the product $a_{ij}\,x_j$ by an integer amount. Let a_{ij}^* equal a_{ij} plus a (positive or negative) integer such that

$$0 \leq a_{ij}^* < 1 \tag{18}$$

then, from (17),

$$-\sum_{j \neq k}^{n} a_{ij}^* x_j + x_{n+1} = -\delta_i \tag{19}$$

where, if all x_j are integers, x_{n+1} is an integer. The term x_{n+1} is a collection of the integers $[b_i]$ and x_k and the integer changes in the $a_{ij}x_j$.

Furthermore, x_{n+1} is a nonnegative integer. Inasmuch as (20)

$$\sum_{j \neq k}^{n} a_{ij}^* x_j \geq 0 \tag{20}$$

it follows from (19) that

$$\delta_i + x_{n+1} \geq 0 \tag{21}$$

But since δ_i is nonnegative and less than 1 and since x_{n+1} is an integer, it follows from (21) that x_{n+1} is a nonnegative integer.

Equation (19) contains the information that the x_j should be integers and (19) is the additional constraint needed to display the integer solution. x_{n+1} is the additional, slack variable. The new equation and the new variable are added to the matrix containing (14) and at least one more iteration is calculated. There is not a direct illustra-

tion of (19) in the appendix table because the problem there allows some variables to remain nonintegers. However, line 13 in the table illustrates the analogous restriction for the mixed integer problem as per equation (36).

Equation (19) is not satisfied at the time it is added to the system. In the first place, not all the x_j are integers as required. In the second place, the slack variable x_{n+1} equals $-\delta_i$ in the current basis which violates the nonnegativity restriction (3). The first computation should be one which re-imposes (3). Then proceed as usual until the second approximation is reached.

With $x_{n+1}<0$ in the current basis, the usual simplex procedure needs a broader than usual interpretation in order to assure that the negative slack is removed from the basis and in order to select an x_j to enter the basis which will not drive some other x_i negative. This is easy to do if the computation is being done with pencil and paper or a desk calculator. Most linear program routines for high speed computers are not designed to handle a negative variable properly. An iteration to remove the negative variable may precede reloading the problem on the machine. Modifications are needed in existing routines to generate the $(m+1)^{st}$ equation and the $(n+1)^{st}$ variable and to handle the negative x_{n+1}. Such modifications are not particularly complicated.

Alternatively, the expanded matrix may be reloaded immediately by multiplying equation (19) by -1 and adding a dummy slack with an indefinitely large negative c value in the criterion function. This procedure adds one equation and two slacks to the original matrix and is illustrated by line 13, table 1 in the appendix.

Sometimes it will happen that the second approximation will satisfy conditions (1) through (4) and display the optimal, integer solution. However, it also sometimes happens that the second approximation contains some unwanted nonintegers as in the example in the appendix. This could result when a_{ij} for a nonbasis integer variable in the first approximation is between zero and one such that $a^*_{ij}=a_{ij}$ as well as when an integer variable is already in the basis. In either event, the new equation would not contain the required information concerning the integer variable. If the second approximation fails to display the required integer solution, generate an additional equation (m+2) and an additional slack variable (x_{n+2}) and proceed as before.

The method of complete integer programming may be outlined in three overtly simple steps:

A. Solve the problem without imposing the integer restriction (4).

B. Add a new slack variable, x_{n+1}, and a new equation of the form:

$$-\sum_{j=1}^{n} f_{ij}x_j+x_{n+1}=-\delta_i \tag{22}$$

where

δ_i=the fractional part of an integer variable which assumed a noninteger value in the current basis

and where

$$f_{ij}=\begin{cases}0 \text{ for all } j \text{ such that } x_j \text{ is in the current basis}\\ a^*_{ij} \text{ for all } j \text{ such that } x_j \text{ is not in the current basis}\end{cases}$$

and where

$$a^*_{ij}=\begin{cases}\text{the fractional part of } a_{ij} \text{ for } a_{ij}\geq 0\\ \text{one minus the fractional part of the absolute value of } a_{ij} \text{ for } a_{ij}<0.\end{cases}$$

C. Continue computing until either the optimal, integer solution appears or until another noninteger optimum appears. In the latter event, repeat steps B and C.

The Partial Integer Solution

The partial, or mixed integer solution of a programming problem requires that some x_j are integers whereas the complete integer solution requires that all x_j are integers (*12*). Consider again equation (14) as the ith of m restrictions in the first approximation, or the noninteger solution of an integer problem. The ith equation may be chosen from the m restrictions in any manner provided that b_i should be an integer in the final solution but is not an integer in the current basis. See line 12 of the appendix table for an example. Defining δ_i as in (16), the additional restriction required for the partial integer solution may be developed by operating on (17)

$$-\sum_{j\neq k}^{n} a_{ij}x_j+[b_i]-x_k=-\delta_i \tag{17}$$

where x_k is the activity in the current basis assigned the value b_i.

Breaking the summation on the left hand size of (17) into two parts according to whether a_{ij} is negative or nonnegative, we have

$$\sum_{j\neq k}^{n} a_{ij}x_j = \sum_{a_{ij}\geq 0} a_{ij}x_j + \sum_{a_{ij}<0} a_{ij}x_j \tag{23}$$

For alternative values of the x_j, the summation on the left hand side of (23) will be either negative or nonnegative. If it is nonnegative, considering that x_k is an integer and that the summation must therefore differ from δ_i by an integer amount, it follows that

$$\sum_{j\neq k}^{n} a_{ij}x_j \geq \delta_i \tag{24}$$

or

$$\sum_{a_{ij}\geq 0} a_{ij}x_j + \sum_{a_{ij}<0} a_{ij}x_j \geq \delta_i \tag{25}$$

which implies

$$\sum_{a_{ij}\geq 0} a_{ij}x_j \geq \delta_i \tag{26}$$

If, on the other hand, the summation in (23) is negative, it follows that

$$\sum_{j\neq k}^{n} a_{ij}x_j \leq \delta_i - 1 \tag{27}$$

or

$$\sum_{a_{ij}\geq 0} a_{ij}x_j + \sum_{a_{ij}<0} a_{ij}x_j \leq \delta_j - 1 \tag{28}$$

which implies

$$\sum_{a_{ij}<0} a_{ij}x_j \leq \delta_i - 1 \tag{29}$$

multiply both sides by -1

$$-\sum_{a_{ij}<0} a_{ij}x_j \geq 1 - \delta_i \tag{30}$$

and multiply both sides by a constant containing δ_i

$$-\left(\frac{\delta_i}{1-\delta_i}\right)\sum_{a_{ij}<0} a_{ij}x_j \geq \delta_i \tag{31}$$

and rearrange

$$\sum_{a_{ij}<0}\left(\frac{\delta_i}{\delta_i - 1}\, a_{ij}\right) x_j \geq \delta_i \tag{32}$$

Combining (26) and (32) leads to the assertion that

$$\sum_{a_{ij}\geq 0} a_{ij}x_j + \sum_{a_{ij}<0}\left(\frac{\delta_i}{\delta_i - 1}\, a_{ij}\right) x_j \geq \delta_i \tag{33}$$

and it is on the inequality (33) that the added restriction to the partial integer programming problem is based. Introducing a nonnegative slack variable into (33) produces the equation

$$-\sum_{a_{ij}\geq 0} a_{ij}x_j - \sum_{a_{ij}<0}\left(\frac{\delta_i}{\delta_i - 1}\, a_{ij}\right) x_j + x_{n+1} = -\delta_i \tag{34}$$

In obtaining (34), the variable x_k was required to be an integer. If x_k is the only integer variable in the problem, equation (34) is the additional restriction required. If some variables in the current basis in addition to x_k are required to be integers, additional restrictions similar to (34) need to be generated. Such information can not be incorporated in (34) because the a_{ij}'s for x_j's included in the current basis are each zero.

If some variables not in the current basis are required to be integers, this information can be introduced into equation (34) in a manner entirely analogous to the adjustments used to transform equation (17) into (19) in the complete integer problem. The procedure is to adjust the left hand side of (33) resulting in a stronger inequality. We shall use the property that the smaller the coefficient of an x_j in (33) the stronger the inequality.

For an integral x_j not in the current basis, its coefficient a_{ij} in (23) may be changed by an integer amount to a_{ij}^*. Such a change in a coefficient would not destroy the fact that when x_k is an integer the summation in (23) differs from δ_i by an integer amount. We wish to change the coefficient of x_j in (23) by an integer amount in a way which results in the inequality in (33) becoming as strong as possible. Among the nonnegative numbers differing from a_{ij} by an integer amount, a coefficient less than 1 leads to the strongest possible inequality. Among the negative numbers differing from a_{ij} by an integer amount, a coefficient greater than -1 leads to the strongest possible inequality.

Therefore, defining a_{ij}^* as a_{ij} plus an integer amount such that

$$0 \leq a_{ij}^* < 1 \tag{18}$$

as before, the choice for a coefficient for x_j in (23) narrows to

$$\begin{matrix} \text{either} & a_{ij}^* \\ \text{or} & a_{ij}^* - 1 \end{matrix} \tag{35}$$

whichever leads to the strongest inequality in (33).

If we choose a_{ij}^* for the coefficient of x_j in (23), x_j has the coefficient a_{ij}^* in (33). On the other hand, if we choose $a_{ij}^* - 1$ as the coefficient in (23),

x_j has the coefficient $\delta_i(a_{ij}^*-1)/(\delta_i-1)$ in (33).

It happens that if $a_{ij}^*=\delta_i$, either choice in (35) would lead to identical results. Consequently, the inequality (33) will be as strong as possible if we choose the coefficient of x_j in (23) for x_j an integer variable according to the rule:

If $a_{ij}^*\leq\delta_i$, let the coefficient in (23) of $x_j=a_{ij}^*$

and

If $a_{ij}^*>\delta_i$, let the coefficient in (23) of $x_j=a_{ij}^*-1$

The method for partial integer programming may be outlined in three overtly simple steps:

(A) Solve the problem without imposing the integer restriction (4)

(B) Add a new slack variable, x_{n+1}, and a new equation of the form (36):

$$-\sum_{j=1}^{n} f_{ij}x_j+x_{n+1}=-\delta_i \tag{36}$$

where

δ_i=the fractional part of an integer variable which assumed a noninteger value in the current basis

and where

$$f_{ij}=\begin{cases} 0 & \text{for all j such that } x_j \text{ is in the current basis} \\ a_{ij} & \text{for all j such that } x_j \text{ is a noninteger variable not in the current basis and } a_{ij}\geq 0 \\ \dfrac{\delta_i}{\delta_i-1}a_{ij} & \text{for all j such that } x_j \text{ is a noninteger variable not in the current basis and } a_{ij}<0 \\ a_{ij}^* & \text{for all j such that } x_j \text{ is an integer variable not in the current basis and } a_{ij}^*\leq\delta_i \\ \dfrac{\delta_i}{\delta_i-1}(a_{ij}^*-1) & \text{for all j such that } x_j \text{ is an integer variable not in the current basis and } a_{ij}^*>\delta_i \end{cases}$$

and where

$$a_{ij}^*=\begin{cases} \text{the fractional part of } a_{ij} \text{ for } a_{ij}\geq 0 \\ \text{one minus the fractional part of the absolute value of } a_{ij} \text{ for } a_{ij}<0 \end{cases}$$

(C) Continue computing until either the optimal, integer solution appears or until another noninteger optimum appears. In the latter event, repeat steps B and C.

Appendix

By way of illustrating the integer programming procedure, let us seek to maximize profits for a farm producing oats, hay, milk, and beef where

$$\pi=15x_1+6x_2+175x_3+76x_4-1000x_5 \tag{37}$$

subject to (38)

$$\begin{array}{rrrrrrrrrrrl} & 1.50x_1 & & +\ 2.00x_3 & +\ .50x_4 & & +x_6 & & & & = & 120 \\ & & .50x_2 & +\ 3.00x_3 & +\ 3.50x_4 & & & +x_7 & & & = & 140 \\ (38) & 6.00x_1 & +3.50x_2 & +70.00x_3 & +40.00x_4 & +400.00x_5 & & & +x_8 & & = & 3500 \\ & & & 1.00x_3 & & -\ 44.00x_5 & & & & +x_9 & = & 0 \\ & & & & +\ 1.00x_4 & +\ 40.00x_5 & & & & & +x_{10} = & 40 \end{array}$$

and to

$$x_i\geq 0 \qquad (i=1, 2, \ldots, 10) \tag{39}$$

and to

(40) x_3, x_4 x_5, x_9, and x_{10} are integers

where the x's are interpreted as follows:

x_1 is tons of oats produced
x_2 is tons of hay produced
x_3 is number of milk cows
x_4 is number of beef cows
x_5 is a (0,1) variable
x_6 is slack acres of crop land
x_7 is slack acres of hay and pasture land
x_8 is slack hours of labor
x_9 is slack milk cow capacity
x_{10} is slack beef cow capacity

Equations (37) to (40) are counterparts to equations (1) to (4), respectively. In this example, we require that livestock units in the solution are integers and we require that the solution may include either milk cows or beef cows but may not include both. The first 2 equations in (38) ensure

that the land used in producing crops and livestock does not exceed total land available on the farm. The third equation in (38) is the labor restriction. This equation indicates that 400 hours of labor are required for each unit of x_5; x_5 is a (0,1) variable which is to have the value of 1 if cows are milked. Thus, if cows are milked, a 400-hour fixed labor requirement is charged against the available supply of labor. At the same time, if cows are milked a fixed cost of $1,000 per year is charged against net revenue for providing services of specialized dairy equipment, according to equation (37). Distributing fixed charges in integer programming is discussed in connection with example (1), page 51.

The two final equations in (38) are counterparts of equation (13) in example (5) page 52. They ensure that the final solution does not contain both beef and dairy cattle. These equations recognize upper bounds for the dairy and beef enterprises of 44 cows and 40 cows, respectively. For the dairy cattle, as many as 60 head could be handled with the 120 acres of cropland available; as many as 46.666 head with the 140 acres of pasture and hay land available; but not more than 44.2857 head with the available supply of labor. Thus, 44 is the largest integer feasible for dairy herd size. Similarly, the hay and pasture land available limits to 40 head the maximum feasible beef herd.

The fourth equation in (38) says if $x_5=1$, the number of cows milked (x_3) plus the slack milk cow capacity (x_9) must total the maximum feasible dairy herd size. On the other hand, if $x_5=0$, x_3 and x_9 must each be zero. The fifth equation in (38) says that if $x_5=0$, the number of beef cows (x_4) plus the slack beef cow capacity (x_{10}) must total the maximum feasible beef herd size. If $x_5=1$, x_4 and x_{10} must each be zero.

Table 1 shows the steps required to reach the mixed integer solution by the ordinary simplex method. The upper section of the table shows the initial basis. The second section shows an approximate, noninteger solution which is obtained by solving equations (37), (38), and (39) without regard to (40). In the first approximation, oats, milk, and beef activities are used to produce a net revenue of $6,820. Three of the basis variables in the approximation, x_3, x_4, and x_5 are supposed to be integers; x_5, the (0,1) variable, has the value 0.9328. This means net revenue fails to reflect about $67 per year of the fixed cost of specialized dairy equipment and it means about 27 hours of the fixed dairy labor requirement are not charged against the fixed labor supply.

Using line (12) in table 1 as the counterpart of equation (14), line (13) is generated to incorporate the information that some variables are to be integers. Line (13) is the counterpart to equation (36). The fractional part of the basis variable in line (12) is $\delta_1=0.9328$. Following the rules for mixed integer programming listed on page 56:

$$f_{12}=a_{12} \qquad =.0022$$

$$f_{16}=\frac{\delta_1}{\delta_1-1}a_{16} \qquad =(-13.8889)(-.0025) \qquad =.0345$$

$$f_{18}=a_{18} \qquad =.0006$$

$$f_{19}=\frac{\delta_1}{\delta_1-1}(a^*_{19}-1) \quad =(-13.8889)(.9614-1.000)=.5355$$

$$f_{110}=\frac{\delta_1}{\delta_1-1}(a^*_{110}-1)=(-13.8889)(.9764-1.000)=.3282$$

To avoid a negative basis variable in line (13), and thereby to facilitate reloading the problem on a highspeed computer, line (13) reflects -1 times equation (36) as per the discussion on page 54 of the text. x_{11} is the counterpart of the slack variable x_{n+1} in equation (36) and x_{12} is a dummy slack. The $1,000 negative coefficient of x_{12} in the criterion function, line (2), is sufficiently large to preclude x_{12} from the final solution.

Line (14) of table 1 shows the derivatives of the profit equation. Elements of line (14) are frequently referred to as C_j-Z_j values in programming literature. Before adding line (13), each derivative was negative indicating that the solu-

680439—63——3

TABLE 1.—*Steps in the solution of a mixed integer programming problem* [1]

		Oats	Hay	Milk	Beef	Dairy equipment	Idle cropland	Idle hay and pasture land	Idle labor	Idle milk capacity	Idle beef capacity				
		Non-integer	Non-integer	Integer	Integer	Integer	Non-integer	Non-integer	Non-integer	Integer	Integer	Non-integer	Non-integer	Non-integer	Non-integer
(1)	X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}
(2)		$15	$6	$175	$76	−$1000	$0	$0	$0	$0	$0	$0	−$1000	$0	−$1000
	INITIAL BASIS														
(3)	120.0000	1.5000		2.0000	.5000		1.0000								
(4)	140.0000		.5000	3.0000	3.5000			1.0000							
(5)	3500.0000	6.0000	3.5000	70.0000	40.0000	400.0000			1.0000						
(6)	0			1.0000		−44.0000				1.0000					
(7)	40.0000				1.0000	40.0000					1.0000				
	FIRST NONINTEGER APPROXIMATION														
(8)	24.3781	1.0000	−.0987				.7794		−.0282	.4146	.7380				
(9)	7.4627		.5174				−.0199	1.0000	.0050	−3.3085	−3.6891				
(10)	41.0448		.0958	1.0000			−.1095		.0274	−.6965	−1.0398				
(11)	2.6866		−.0871		1.0000		.0095		−.0249	1.5423	1.9453				
(12)	.9328		.0022			1.0000	−.0025		.0006	−.0386	−.0236				
(13)	.9328		.0022				.0345		.0006	.5355	.3282	−1.0000	1.000		
(14)			$1.69				$31.07		−$1.23	$495.48	$327.69	−$1000.00			
	SECOND APPROXIMATION														
(15)	23.6559	1.0000			−1.0753		.7527				−.5914	−2.3226	2.3226		
(16)	13.2258				5.6878		.1935	1.0000	−.1429		4.0265	10.2028	−10.2028		
(17)	42.2581			1.0000	1.0565		−.0645				.4435	1.7419	−1.7419		
(18)	0		1.0000		−10.7143				.2857		−10.7143	−30.8571	30.8571		
(19)	1.0000				.0250	1.0000					+.0250				
(20)	1.7419				.0435		.0645			1.0000	.6565	−1.7419	1.7419		
(21)	.2581				.0565		.0224				.1427	1.7419	.6057	−1.0000	1.0000
(22)					$53.03		$22.24		−$1.71		$113.24	$1657.05	−$309.45	$1000.00	
	MIXED INTEGER SOLUTION														
(23)	15.0000	1.0000			−2.9687						−7.0822	−60.7483	−18.0000	33.5407	−33.5407
(24)	11.0000				5.2009			1.0000	−.1429		2.3574	−4.8210	−15.4286	8.6248	−8.6248
(25)	43.0000			1.0000	1.2187						1.0000	6.7500		−2.8749	2.8749
(26)	0		1.0000		−10.7143				.2857		−10.7143	−30.8571	30.8571		
(27)	1.0000				.0250	1.0000					.0250				
(28)	1.0000				−.1187					1.0000	1.0000	−6.7500		2.8749	−2.8749
(29)	11.5000				2.5156		1.0000				8.6235	77.6228	27.0000	−44.5612	44.5612
(30)					−$3.46				−$1.71		−$96.34	−$84.88	−$915.14	$0.	−$1000.00

[1] Numbers in this table are rounded to 4 decimal places, but computations were made carrying more decimals, so some computations may appear not to add up.

tion was optimal given the restrictions imposed. With the addition of line (13), several of the derivatives are positive indicating that, under the added restriction, some other feasible farm organization than the current one would be more profitable.

The third section of table 1 shows the second approximation obtained by seeking to maximize profits subject to the restrictions represented by lines (8) to (13) of the table. In the second approximation, oats and milk activities are used to produce a net revenue of $6,750, x_5 is an integer as required. However, x_3 and x_9, also in the basis, are not integers. Evidently the counterpart of inequality (33) was not strong enough to produce the desired integer solution—the process must be repeated.

Using line (17) as the counterpart to equation (14), the rules for mixed integer programming were used to generate line (21). Line (22) shows the $C_j - Z_j$ values. Solving the expanded, 7 by 14 matrix led to the solution in the lower section of the table in which oats and milk activities are used to produce a net revenue of $6,750, the same revenue as obtained in the previous approximation. The derivatives in line (30) are each negative, indicating that this is the mixed integer solution which maximizes (37) subject to (38, 39, and 40).

Literature Cited and Selected References

(*1*) Beale, E. M. L.
1958. A method of solving linear programming problems when some but not all of the variables must take integral values. Statistical Techniques Research Group Technical Report No. 19. Princeton, N.J. July.

(*2*) Bennett, J. M., and Dakin, R. J.
1961. Experience with mixed integer linear programming problems. Technical Report No. 18. Basser Computing Department. University of Sydney, Sydney, Australia. Oct.

(*3*) Candler, Wilfred, and Manning, Richard.
1961. A modified simplex solution for problems with decreasing average costs. Journ. Farm Econ. Menasha, Wisc. Nov. 43: 859–875.

(*4*) Chou, Tao-hsiung.
1961. Mixed integer programming with application of dynamic and non-linear models for agriculture. Ph. D. Thesis, Iowa State Univ., Ames, Iowa.

(*5*) Chou, T. H., and Heady, E. O.
1961. Applications in integer programming. Canadian Jour. Agr. Econ. Ottawa, Canada. 9: 54–67.

(*6*) Dantzig, G. B.
1957. Discrete-variable extremum problems. Operations Research, 5, Baltimore, Md. 15: 266–277.

(*7*) ———
1958. On integer and partial integer linear programming problems. The Rand Corp. Santa Monica, Calif. Paper P–1410. June.

(*8*) ———
1960. On the significance of solving linear programming problems with some integer variables. Econometrica, New Haven, Conn. 28: 30–44. Jan.

(*9*) Edwards, Clark
1961. Shortcomings in programmed solutions to practical farm problems. Jour. Farm Econ. Menasha, Wis. 43: 393–401. May.

(*10*) ———
1962. Non-linear programming and non-linear regression procedures. Jour. Farm Econ. Menasha, Wis. 44: 100–114. Feb.

(*11*) Gomory, R. E.
1958. An algorithm for integer solutions to linear programs. Princeton-I.B.M. Mathematics Research Project, Technical Report No. 1. Princeton, N.J. Nov.

(*12*) ———
1960. An algorithm for the mixed integer problems. The Rand Corp.

Santa Monica, Calif. Research Memorandum RM–2597. July.

(*13*) ———
1958. OUTLINE OF AN ALGORITHM FOR INTEGER SOLUTIONS TO LINEAR PROGRAMS. Bulletin of the American Mathematical Society. Providence, R.I. Vol. 64, No. 5. Sept.

(*14*) GOMORY, R. E., AND BAUMOL, WILLIAM J.
1960. INTEGER PROGRAMMING AND PRICING. Econometrica. New Haven, Conn. 28: 521–550. July.

(*15*) HIRSCH, WARREN M., AND DANTZIG, G. B.
1954. THE FIXED CHARGE PROBLEM. The Rand Corp., Santa Monica, Calif. Paper P–648. Dec.

(*16*) LAND, A. H., AND DOIG, A. G.
1960. AN AUTOMATIC METHOD OF SOLVING DISCRETE PROGRAMMING PROBLEMS. Econometrica. New Haven, Conn. 28: 497–520. July.

(*17*) MARKOWITZ, H. M., AND MANNE, A. S.
1957. ON THE SOLUTION OF DISCRETE PROGRAMMING PROBLEMS. Econometrica. New Haven, Conn. 25: 84–110. Jan.

(*18*) MUSGRAVE, WARREN F.
1962. A NOTE ON INTEGER PROGRAMMING AND THE PROBLEM OF INCREASING RETURNS. Jour. Farm. Econ. Menasha, Wis. 44: 1068–1076. Nov.